MoSKito Inspect REST API

Since MoSKito 2.2.0 MoSKito Inspect has an integrated RESTful interface, that is parallel to the user interface. This document defines the usage of the MoSKito Inspect RESTful interface.

The API has been refactored in version 2.6.0. The old API can be found here:MoSKito Inspect pre 2.6 REST API.

- General
  - Request format
  - Response format
    - Common response objects
      - Accumulator Definition
      - Threshold Definition
      - Threshold Status
      - Dashboard Definition
      - DashboardChartDefinition
      - Gauge
      - Dashboard Chart
      - Dashboard
  - Cases

General

Request format

Requests to the REST Interface should have the form:

```
http://server:port/<app>/<restpath>/<case-url>/<parameters>
```

Not all parts of the path must be always present.

Parameter	
server	Name of the server, for example */localhost* or *www.mydomain.com*
port	Port of the server. For example 8088.
app	Name of your application in which MoSKito has been deployed. For standalone MoSKito Inspect it will be *moskito*
restpath	Path of the Jersey servlet. We use *moskito-inspect-rest* per default.
case-url	The specific url for the case. The url is defined in the table below.
parameters	Get parameters if required.

Response format

The response can be provided as XML or JSON. Note, that we use mainly JSON for our applications, so XML format may be a little bit less tested. The response format is defined by *Accept* header.

- application/json - for JSON
- application/xml - for XML

All replies are packed in a general response object.

JSON Example:

```
{
    "success": true,
    "results": {
        "accumulators": [
            {
                "id": "46",
                "name": "CPU Time 1h",
                "path": "OS.OS.CPU Time/1h/MILLISECONDS",
                "numberOfValues": 0,
                "lastValueTimestamp": "none",
                "maxNumberOfValues": 275
            },
            {
                "id": "44",
                "name": "CPU Time 1m",
                "path": "OS.OS.CPU Time/1m/MILLISECONDS",
                "numberOfValues": 0,
                "lastValueTimestamp": "none",
                "maxNumberOfValues": 275
            }
       ]
    }
}
```

The general response object contains at least two fields:

- Status - could the request be processed correctly. If the status is *false* you should throw away the response.
- Results - a dictionary (hashmap) with result objects. The result objects differ from case to case.

The result objects for each case are defined in the Cases section.

Common response objects

Some response objects are used at multiple locations. For easy of documentation we document them here, and refer to them in the later documentation.

Accumulator Definition

An accumulator represents a MoSKito accumulator definition object, not the data collector for charting. It is represented byAccumulatorDefinitionAO. This object contains the definition of the Accumulator without its current data state.

```
            {
                "id": "<accumulator id>",
                "name": "<accumulator name",
                "path": "<internal moskito path>",
                "numberOfValues": <Current amount of accumulated values>,
                "lastValueTimestamp": "<Timestamp of last accumulated value>",
                "maxNumberOfValues": <Maximum amount of values that this accumulator will accumulate>
            }
```

Example:

```
            {
                "id": "46",
                "name": "CPU Time 1h",
                "path": "OS.OS.CPU Time/1h/MILLISECONDS",
                "numberOfValues": 3,
                "lastValueTimestamp": "2015-04-14T15:39:05,417",
                "maxNumberOfValues": 275
            },
```

Attributes:

- id - The internal accumulator id. Use it to reference this accumulator
- name - The human readable name of the accumulator, that has been provided in configuration or at creation time.
- path - Definition of components that accumulator is refering too in form **ProducerId.StatName.ValueName/Interval/TimeUnit**
- numberOfValues - Amount of values that have been collected already.
- lastValueTimestamp - timestamp in ISO format, when the last value has been collected.
- maxNumberOfValues - maximum number of values that accumulator is allowed to collect. If the number of collected values exceeds this limits, older values are dropped.

Threshold Definition

Threshold definition describes how the Threshold has been defined.

```
{
                "id": "3",
                "name": "ThreadCount",
                "producerName": "ThreadCount",
                "statName": "ThreadCount",
                "valueName": "Current",
                "intervalName": "default",
                "descriptionString": "ThreadCount.ThreadCount.Current/default/MILLISECONDS",
                "timeUnit": "MILLISECONDS"
}
```

Attributes:

- id - id of the thresholds
- name - user given name of the threshold
- producerName - name of the associated producer (*ProducerId*)
- statName - name of the associated *StatValue* in the producer.
- valueName - name of the associated *Value* in the StatValue.
- intervalName - name of the interval (1m, 5m etc)
- descriptionString - a one string description of the threshold
- timeUnit - associated *TimeUnit*.

Threshold Status

Threshold Status is the current status of the Threshold:

```
            {
                "name": "ThreadCount",
                "colorCode": "green",
                "timestamp": "2015-04-14T18:11:00,706",
                "description": "ThreadCount.ThreadCount.Current/default/MILLISECONDS",
                "value": "23",
                "id": "3",
                "flipCount": 1,
                "timestampInMillis": 1429027860706
            },
```

Attributes:

- name: name of the Threshold (same as in the ThresholdDefinition)
- id: id of the Threshold (same as in the ThresholdDefinition)
- colorCode: current color of the Threshold. One of: green, yellow, orange, red, purple, none.
- timestamp: ISO timestamp of the last Threshold flip (change).
- value: last value of the threshold.
- flipCount: number of flips of this thresholds sofar. A flip is a color change.
- timestampInMillis: timestamp in milliseconds since 1970 for calculations.

Dashboard Definition

Dashboard definition object describes configured dashboard. It doesn't contain any data.

```
            {
                "name": "Dashboard name",
                "gauges": [String,...],
                "thresholds": [String,...],
                "charts": [DashboardChartDefinition, ...]
            }
```

DashboardChartDefinition

```
{
  "caption": "Memory",
  "accumulatorNames": [
    "OldGenFree MB 1m",
    "OldGenUsed MB 1m",
    "PermGenFree MB 1m"
  ]
}
```

Example for Dashboard and DashboardChart definitions:

```
{
    "success": true,
    "results": {
        "definitions": [
            {
                "name": "Example Dashboard",
                "gauges": [
                    "Blocked",
                    "Memory",
                    "Running",
                    "Sessions"
                ],
                "thresholds": [
                    "ThreadCount",
                    "OldGenFree",
                    "PermGenFree"
                ],
                "charts": [
                    {
                        "caption": "Memory",
                        "accumulatorNames": [
                            "OldGenFree MB 1m",
                            "OldGenUsed MB 1m",
                            "PermGenFree MB 1m"
                        ]
                    },
                    {
                        "caption": "Threads",
                        "accumulatorNames": [
                            "ThreadCount",
                            "ThreadStateBlocked-1m",
                            "ThreadStateRunnable-1m",
                            "ThreadStateTimedWaiting-1m",
                            "ThreadStateWaiting-1m"
                        ]
                    },
                    {
                        "caption": null,
                        "accumulatorNames": [
```

```
                        "URL REQ 1m"
                    ]
                },
                {
                    "caption": null,
                    "accumulatorNames": [
                        "URL Time 1m"
                    ]
                },
                {
                    "caption": null,
                    "accumulatorNames": [
                        "URL AVG 1m"
                    ]
                },
                {
                    "caption": null,
                    "accumulatorNames": [
                        "SessionCount Cur Absolute"
                    ]
                },
                {
                    "caption": null,
                    "accumulatorNames": [
                        "CPU Time 1m"
                    ]
                }
            ]
        },
        {
            "name": "Empty Dashboard",
            "gauges": [],
            "thresholds": [],
            "charts": []
        }
    ]
}
```

Gauge

A gauge is represented by following value:

```
            {
                "name": "Blocked",
                "caption": "Blocked Threads",
                "min": "0",
                "current": "0",
                "max": "23",
                "complete": true
            },
```

- Name of the gauge
- Human friendly caption
- Configured min value
- Configured or calculated max value
- Calculated current value
- Complete flag. If it is false, the gauge doesn't deliver data yet.

Dashboard Chart

```
DashboardChart:
{
  "caption": "Memory",
  "chartData": [DashboardChartData, ...]
  "lineNames": [String,...]
}

DashboardChartData
{
  "name": "-",
  "data": [ChartDataEntity]
}

ChartDataEntity
{
"values": [LineValues as String, ...]
"timestamp": "1429198196000",
"numericTimestamp": 1429198196000
}
```

Example of DashboardChart


```
{
    "caption": "Memory",
    "chartData": {
        "name": "-",
        "data": [
            {
                "values": [
                    "141",
                    "29",
                    "6"
                ],
                "timestamp": "1429198196000",
                "numericTimestamp": 1429198196000
            },
            {
                "values": [
                    "141",
                    "29",
                    "0"
                ],
                "timestamp": "1429198256000",
                "numericTimestamp": 1429198256000
            },
            {
                "values": [
                    "141",
                    "29",
                    "0"
                ],
                "timestamp": "1429198316000",
                "numericTimestamp": 1429198316000
            },
            {
                "values": [
                    "141",
                    "29",
                    "0"
                ],
                "timestamp": "1429198376000",
                "numericTimestamp": 1429198376000
            },
```

```
                        {
                            "values": [
                                "141",
                                "29",
                                "0"
                            ],
                            "timestamp": "1429198436000",
                            "numericTimestamp": 1429198436000
                        },
                        {
                            "values": [
                                "141",
                                "29",
                                "0"
                            ],
                            "timestamp": "1429198496000",
                            "numericTimestamp": 1429198496000
                        },
                        {
                            "values": [
                                "141",
                                "29",
                                "0"
                            ],
                            "timestamp": "1429198556000",
                            "numericTimestamp": 1429198556000
                        },
                        {
                            "values": [
                                "141",
                                "29",
                                "0"
                            ],
                            "timestamp": "1429198616000",
                            "numericTimestamp": 1429198616000
                        }
                    ]
                },
                "lineNames": [
                    "OldGenFree MB 1m",
                    "OldGenUsed MB 1m",
                    "PermGenFree MB 1m"
                ]
            },
```

Dashboard

The dashboard object represent a state of a dashboard including all relevant data.

```
  "results": {
       "dashboard": {
           "gauges": [ Gauge, ...]
           "thresholds": [ ThresholdStatus, ...]
           "charts": [ DashboardChart, ...]
                       }
       }
```

Cases

Section	Case	Case-URL	Method	Parameters	Example URL	Response	
Accumulators							
	List all accumulators	/accumulators	GET		`http://<server>:<port>:<app>/moskito-inspect-rest/accumulators`	`"accumulators": [ AccumulatorDefinition, AccumulatorDefinition, ...]`	
	Get a single accumulator	/accumulators/{id}	GET	• id - accumulator id	`http://<server>:<port>:<app>/moskito-inspect-rest/accumulators/22`	`"accumulator": AccumulatorDefinition, "chartData": ChartData`	
	Delete an accumulator	/accumulators/{id}	DELETE	• id - accumulator id	`http://<server>:<port>:<app>/moskito-inspect-rest/accumulators/22`	none. Example: `{ "success": true, "results": {} }`	Warning, this will actually **drop** all collected data.

	Create an accumulator	/accumulators	POST	POST data **JSON** or **XML**:
				
				`{ "name": "TestAccumulator", "producerId": "ThreadCount", "interval": "1m", "unit": "MILLISECONDS", "statName": "ThreadCount", "valueName": "current" }`
					`http://<server>:<port>/<app>/moskito-inspect-rest/accumulators`	`"created": AccumulatorDefinition`
						
						Example:
						
						`{ "success": true, "results": { "created": { "id": "47", "name": "TestAccumulator", "path": "ThreadCount.ThreadCount.current/1m/MILLISECONDS", "numberOfValues": 0, "lastValueTimestamp": "none", "maxNumberOfValues": 0 } } }`
							Note, if the name is already used, a new name is selected, which is old name plus **-number**.

	Returns a chart of multiple accumulators	/accumulators/combined?id={id}&id={id}...	GET	• id - multiple ids of accumualtors	`http://<server>:<port>/<app>/moskito-inspect-rest/accumulators/combined?id=22&id=23`	Example: `{ "success": true, "results": { "chart": { "data": [ { "values": [ "18", "3" ], "timestamp": "22:52", "isoTimestamp": "2015-06-27T22:52:00,000",`	

						"numericTimestamp": 1435438320000 }, { "values": ["18", "6"], "timestamp": "22:53", "isoTimestamp": "2015-06-27T22:53:00,000", "numericTimestamp": 1435438380000 }, { "values": ["18", "5"], "timestamp": "22:54", "isoTimestamp": "2015-06-27T22:54:00,000", "numericTimestamp": 1435438440000 }], "names": ["ThreadCount", "ThreadStateRunnable-1m"] } } }	
	Returns a chart of multiple accumulators, normalized to 100%	/accumulators/normalized?id={id}&id={id}...	GET	same as combined	same as combined	same as combined	
Thresholds							

	Definitions of the Thresholds	/thresholds /definitions	GET		`http://<server>:<port>:<app>/moskito-inspect-rest /thresholds /definitions`	`"definitions": [ThresholdDefinition, ...]`	
	Current statuses of the Thresholds.	/thresholds /statuses	GET		`http://<server>:<port>:<app>/moskito-inspect-rest /thresholds/statuses`	`"statuses": [ThresholdStatus, ... ]`	
	Returns both, definitions and statuses of the Thresholds to save bandwidth.	/thresholds	GET		`http://<server>:<port>:<app>/moskito-inspect-rest /thresholds`	`"definitions": [ThresholdDefinition, ...], "statuses": [ThresholdStatus, ... ]`	
	Delete a threshold	/thresholds/{id}	DELETE				
	Create a new Threshold	/thresholds	POST				
Alerts							
	Get current alerts. An alert is triggered if the Threshold status is changed.	/alerts	GET	-	`http://<server>:<port>:<app>/moskito-inspect-rest/alerts`	`"alerts": [Alert, Alert..]`	
Status							
	Worst application status	/status	GET		`http://<server>:<port>:<app>/moskito-inspect-rest/status`	"status": "COLOR" Example: `{ "success": true, "results": { "status": "GREEN" } }`	
	Worst application status from submitted	/status	POST	`{ "thresholdNames": ["ThreadCount"] }`	`http://<server>:<port>:<app>/moskito-inspect-rest/alerts`	Same as GET /status	
Dashboards							

	List of Dashboard definitions	/dashboards	GET		`http://<server>:<port>:<app>/moskito-inspect-rest/dashboards`	`"dashboards": [DashboardDefinition, ...]`	
	Retrieval of Dashboard data.	/dashboards/{name}	GET	• name of the dashboard	`http://<server>:<port>:<app>/moskito-inspect-rest/dashboards/Example%20Dashboard`	`"dashboard": Dashboard`	
Producers							
	Get list of all producers and their cumulated data	/producers/{intervalname}/{timeunit}	GET	• intervalname - name of the interval for the data, one of 1m, 5m, 15m etc • timeunit - timeunit for duration related values like Total or AVG, for example MILLISECONDS, MICROSECONDS etc			
	Get the data for a single producer	/producers/{id}/{intervalname}/{timeunit}	GET	• id - the id of the producer • intervalname - name of the interval for the data, one of 1m, 5m, 15m etc • timeunit - timeunit for duration related values like Total or AVG, for example MILLISECONDS, MICROSECONDS etc			
	Return available subsystems	/producers/categories	GET				
	Return available categories	/producers/subsystems	GET				
	Return producers by category	/producers/byCategory/{category}/{intervalname}/{timeunit}	GET	• category - selected category • intervalname - name of the interval for the data, one of 1m, 5m, 15m etc • timeunit - timeunit for duration related values like Total or AVG, for example MILLISECONDS, MICROSECONDS etc			
	Return producers by subsystem	/producers/bySubsystem/{subsystem}/{intervalname}/{timeunit}	GET	• subsystem - selected subsystem • intervalname - name of the interval for the data, one of 1m, 5m, 15m etc • timeunit - timeunit for duration related values like Total or AVG, for example MILLISECONDS, MICROSECONDS etc			
Values							

	Retrieve singe value	/value/{producerId}/{statName}/{valueName}/{interval}/{timeUnit}	GET	• producerId - id of the producer • statName - of the selected stat • valueName - the name of the value • interval - name of the interval • timeunit - timeunit like MICROSECONDS or MILLISECONDS, only for time dependent values	`http://<server>:<port>/<app>/moskito-inspect-rest/value/ShopService/placeOrder/Req/default/MILLISECONDS`		
	Retrieve multiple values		POST		`http://<server>:<port>/<app>/moskito-inspect-rest/value`		
Version							
	Return version information of the server	/version	GET	-	`http://<server>:<port>/<app>//moskito-inspect-rest/version`		`{ "success": true, "results": { "version": { "fileTimestamp": "2015-06-28T00:25:50,000", "version": "2.6.2-SNAPSHOT", "group": "net.anotheria", "artifact": "moskito-webui" } } }`